ACTION SUMMARY MODULE

TECHNICAL REQUIREMENTS

1.1 Introduction

This document is the technical response to the SWSS Foster Care Requirement for Action Summary (FIA-69). It will describe how the development team will implement the changes and additions to SWSS Childrens to effect the requirements.

This document is also to be used as a tool by the development team when coding the solution or maintaining it in the future. Thus this document is likely to be updated during the lifecycle of the SWSS project. Versions of this document will be maintained in PVCS, and the reader should be aware that multiple printed versions may exist.

1.2 Module Description

The Action Summary (FIA-69) module is used whenever there is "action" on a case, e.g., the child moves or the parents move. Because identifying information is requested, this form also serves as notice to the courts of changes in placements, in parent's living situation and the foster care worker/agency. The Foster Care Action Summary meets licensing requirements for Replacement Documentation and Termination from Foster Care Summary when a child moves. The Foster Care Action Summary is also used for the Closing Summary for a foster care case.

1.3 Requirements

1.3.1 Process Description

FIA direct service staff or monitoring workers will be entering the data. Data for the Foster Care Action Summary is collected in several other modules within SWSS. When data is changed, the module must ask if the user wants to go to report generation and print the FIA-69 or if there are additional changes which have not been entered and will affect the information on the FIA-69. If the user answers 'yes' s/he will be taken to Report Generation. If the answer is 'no', the user can proceed with data entry in the application.

1.3.2 Functional Requirements

List any requirements this module implements that are not specifically covered in the User Requirements for this module. For example, Soundex must return the Person ID of a person and write it in the SWSS_INI.ini so that CaseReg can "resurrect" that person with the new case, or Funding Determines the eligibility used in placement and updates the SWSS_CASE.Latest_Funding_Determination

The Foster Care Action Summary is printed from the Report Generation Module when any of the following actions occur:

- The child moves to a new foster home
- The foster care worker or the child placing agency changes
- The foster care case is closed
- There is a change in the parent(s)' living situation
- The child leaves family foster care and goes to any other living arrangement.

1.3.3 Business Events

The Member Module collects the information on parent living situation (address) changes and must allow access to the FIA-69.

The Placement Module collects the information on child moves and child placing agency changes and must allow access to the FIA-69.

The Case Closing Module must access the FIA-69 to record social work contacts before the foster care case is closed.

The Case Listing Module must access the FIA-69 to record primary worker changes.

Every time a new placement record is entered, the user will be reminded to complete the FIA-69, Foster Care Action Summary.

1.3.4 List of Program Units

This would be stuff like common VB code called, the number of VB .BAS modules and form modules in the current application. Also list the Stored Procedures called. Show a "Structure Diagram" of which VB procedure calls a stored procedure or another VB procedure. Also show which stored procedures call other stored procedures.

FORM	FUNCTION	SUB FUNCTION	DATABASE CALLS
FrmACTN	Form_Activate	ExtractINI_Info	
	fGetAcceptDate		action_summary_data.GetAcceptDate
	are_there_current_comments		action_summary_data.get_action_summary_comments
	clean_up_action_summary		placement.clean_up_action_summary
	check_prov_info_shared	maintain_prov_info_share	
	maintain_shared_info		placement.maintain_shared_info
	insert_action_comment		longpkg.INSERT_REC
	update_action_comment		placement.update_action_comment
	action_comments	update_action_comment write_comment_id_out insert_action_comment	
	placement_reasons	add_placement_reasons	
	cmdContinue_Click()	validate_date_fields get_current_placement clean_up_action_summary maintain_shared_info check_prov_info_shared placement_reasons action_comments print_action_summary	
	Form_Load	SetDeviceIndependent	
ACT_SUM.bas	find_docket_number		action_summary_data.find_docket_number
	get_child_case_info		miscpkg.get_child_case_info
	print_social_work_contacts		CONTACTS.GET_CONTACTS
	delete_activity		action_summary_data.delete_activity
	get_mother_address_history		miscpkg.get_mother_addr_hist
	get_father_address_history		miscpkg.get_father_addr_hist
	get_rept_types		action_summary_data.get_rept_types
	get_worker_data		action_summary_data.get_worker_data
	print_action_summary	get_rept_types get_child_case_info	

		find_docket_number get_worker_data get_provider_data get_info_shared get_plcmnt_end get_action_summary_comments get_provider_info_share get_mother_address_history get_father_address_history delete_activity print_social_work_contacts	
	get_info_shared		action_summary_data.get_info_shared
	get_plcmnt_end		action_summary_data.get_plcmnt_end
	get_action_summary_comments		SQL = "SELECT text,comment_type FROM comments " _ & " WHERE log_id = " & gtLogNumber _ & " AND county_no = " & gtCaseCounty _ & " AND comment_type IN ('P1','P2','P3','P4','P5','P6','P7','P9')"
	get_provider_data	get_current_provider get_previous_provider	action_summary_data.find_provider_data
	get_current_provider		placement.get_placement_provider
	get_previous_provider		placement.get_placement_provider
	get_provider_info_share		action_summary_data.get_provider_info_share
	get_current_placement		action_summary_data.get_current_placement
	add_placement_reasons		placement.placement_end_reasons
	print_social_work_contacts		CONTACTS.GET_CONTACTS
	maintain_prov_info_share		placement.maintain_prov_info_share
FrmMain	Form_Load	ExtractINI_Info SetDeviceIndependentWindow get_rept_types	
	CmdContinue_Click()	print_action_summary	

1.3.6 Report (output) Images

Print out versions of each output report generated by the module. For each image, explain its usage.

Report Figure 1. FIA 69 Report

Foster Care Structured Decision Making
Foster Care Action Summary
Michigan Family Independence Agency

FC Case #:
FC Case Name:
FIA FC Worker #:
FIA FC Worker Name:
PS Case Name:
PS Case #:
Court ID#:
POS Agency Name:
POS Agency Worker Name:

Date Completed:

I. Type of Action (Check One) **Effective Date:**

___Child Replacement
___Parent Move
___Caseworker Change
___Termination of Family Foster Care Placement
___ Foster Care Case Closing

II. Child Information

Name: FIA Case Number:
Sex: Docket Number:
Race: Funding Source:
D.O.B:

III. Caseworker Change

(Former) Caseworker's Name: Load Number:
Phone #:

New Caseworker's Name: Load Number:
Phone #:

IV. Parent or Child Move Summary

Parent Name:
Prior Address: New Address:

Old Telephone: New Telephone:

Child Name:
Moved From: Moved To:

Old Phone: New Phone:
Foster Home Provider (MPS#): Foster Home Provider (MPS#):
Primary Provider (MPS#): Primary Provider (MPS#):

Complete Section A or B*

A: Foster care continues to be appropriate for the following reason(s):
(Check as many as apply)

___Children remain at risk if returned to the parental home
___No interested relatives for placement
___No appropriate relative placement

B: Reason for replacement or termination from foster care:
(Check as many as apply)

Requesting the move: ___ Agency ____ Foster Parent ____ Child ____ Court

Planned move, at least 72 hours notice to the foster family and the child (unless court ordered): ___

Unplanned moved: (Can only be Foster Parent request and/or CPS complaint) ___

____ Behavioral problems of the child
____ Emergency or temporary placement
____ Residential Placement
____ Independent Living
____ Placed in adoptive home
____ Complaint against foster parent/caregiver – Agency investigation (Check at least one)
____ CPS Investigation ____ Licensing Investigation
____ Unsuitable relative home
____ Other
____ Foster Parent crisis
____ Placement with relatives
____ Return home
____ AWOL
____ Placed with siblings

Replacement preparation and/or termination appropriate to the child's capacity to understand; give a description on how the worker prepared the child and foster parent for the move:

V. Information related to the care and supervision of the child or termination from Foster Care with:

____ Mother	Date:	via: ____ letter	____ face to face	____ telephone
____ Father	Date:	via: ____ letter	____ face to face	____ telephone
____ New Care Giver	Date:	via: ____ letter	____ face to face	____ telephone
____ FIA/Referring worker	Date:	via: ____ letter	____ face to face	____ telephone
____ Kinship Family Members	Date:	via: ____ letter	____ face to face	____ telephone

Information shared with new Care Giver(s) includes: (Check as many as apply)

____ Assigned caseworker
____ School Records
____ Reason(s) child removed
____ Behavior management
____ Description of Behavioral Characteristics and Needs
____ Medical/Dental/Psychological Needs and/or Files
____ Case plan

_____ Visitation expectations/schedule
_____ Interactions with parents/siblings
_____ Consent to treatment card
_____ Abuse/neglect history
_____ School enrollment form

VI. For Termination of Family Foster Care or Case Closure

1. Reason for closure:

2. Summarize services that were provided to the child and family:

3. Summarize services currently being provided to the child and family:

4. List services and needs which still need to be provided to the child and family:

5. Medical information to be given to parents or next provider: _____ Yes _____ No

6. Was termination or closure explained to all parties? _____ Yes _____ No

7. If termination is unplanned, summarize the reasons and circumstances surrounding the termination:

FIA Foster Care Worker: _____ **Date**:_____

Foster Care Supervisor:_____ **Date:** _____

FIA-69 02/00(SWSS facsimile)
Last printed 3/9/2010 4:40:00 PM

Report Figure 2. Social Work Contacts

Foster Care Structured Decision Making
Foster Care Action Summary
Michigan Family Independence Agency

FC Case #:
FC Case Name:
FIA FC Worker #:
FIA FC Worker Name:
PS Case Name:
PS Case #:
Court ID#:
POS Agency Name:
POS Agency Worker Name:

Date Completed:

Report Period Covered: to

Social Work Contacts Since Last USP:

This attachment to the FIA-69 only prints for a closed case.

1.3.7 Data Elements

For every element on a screen output report:

- Map each data element displayed, printed, or entered to the database table and field, such as the ex done by SDM that Paula mentioned in the team meeting (p:\users\share\servwork\SWSS\templates\ReqTemplates\DataDefinitions).xls
- Make a specific reference to the SWSS Schema data dictionary, which ought to be available any day now.
- List and discuss any specific validation routines, constraints, or dependent data validations (like legal status and living arrangement) that are not in the data dictionary. You can check (and copy from) the User Requirements Data Element Description for these type of validations.
- Explain the instancing of this data element in laymen's terms. This is implied in the table name, usually, such as the "Case_Person" table refers to an instance of a (group) person record in a particular case, and the "Group_Person" table refers to an instance of a person in a particular sibling group. Go ahead and say it like that, as it applies. This includes "historical" data, such as Medicaid_History, which is an instance of medicaid data over time.

1.3.8 Integration with Existing System

How does this module integrate with SWSS. Is it a selection from the main menu? Are there specific things needed in the SWSS_INI.ini? Are there short cut keys or menu selections? Can it be called directly from another module, during which time the calling module shows a blank screen or mess with the task bar, and does it need to update something for that calling module?

The FIA-69 Action Summary Report is located on the Report Generation menu. It may only be access through Report Generation, Member Information(parent address change), Case Closing, Case Listing(caseworker change) and Placement(replacement or termination). The items needed from the SWSS_INI.ini include; gtLogNumber, gtCaseStateDesc, gtCaseNumber, gtCaseName, gtWorkerLoadNumber, gtProgramCode, sCaseStatus. There are short cut keys on only the Previous, Next, Cancel and Continue buttons. This module contains the Common Menu Bar. This module updates nothing for any other module.

1.3.9 Module Dependencies

If this module depends on other modules, or if other modules depend on this module, to an extent beyond what has been described in the "Integration with Existing System" section above, such as Medicaid requiring both Placement and Funding data, list and explain those dependencies here.

Same as mentioned in Section 1.3.8 Integration with Existing System.

1.3.10 Database Subject Area

```
PACKAGE      PLACEMENT
AS
/*************************************************************
        DEFINE TABLES USED IN THIS PACKAGE
*************************************************************/

*****************************************************************************
        GET A SPECIFIC PROVIDER
This procedure reads the provider,provider_address,address, provider_phone,phone,
provider_person,provider_person_race
*****************************************************************************
        PROCEDURE get_placement_provider
                (i_provider_id                  IN      provider_person.provider_id%TYPE,
                i_provider_type                 IN      provider_person.provider_type%TYPE,
                i_placing_agency_id         IN    provider_case_person.placing_agency_id%TYPE,
                i_placing_agency_type       IN    provider_case_person.placing_agency_type%TYPE,
                o_cpa_name                      OUT   provider.provider_name%TYPE,
                o_cpa_address_id            OUT   provider_address.address_id%TYPE,
                o_cpa_line1                     OUT    address.line1%TYPE,
                o_cpa_line2                     OUT    address.line2%TYPE,
                o_cpa_city                      OUT    address.city%TYPE,
                o_cpa_state_code            OUT        address.state_code%TYPE,
                o_cpa_zip                       OUT    address.zip%TYPE,
                o_cpa_zip_plus                  OUT    address.zip_plus%TYPE,
                o_cpa_country                   OUT    address.country%TYPE,
                o_cpa_phone_id              OUT        provider_phone.phone_id%TYPE,
                o_cpa_phone_number          OUT        VARCHAR2,
                o_cpa_province                  OUT   address.province%TYPE,
                o_prov_phone_id             OUT   provider_phone.phone_id%TYPE,
                o_prov_phone_no             OUT    VARCHAR2,
```

```
                o_prov_alt_phone_id        OUT   provider_phone.phone_id%TYPE,
                o_prov_alt_phone_no        OUT   VARCHAR2,
                GetProvCursor                IN OUT Get_Prov_Cursor);
*************************************************************************************************
```

■ REMOVE OLD ACTION SUMMARY DATA - HISTORY IS NOT KEPT ON THIS DATA

Because history is not needed on the Action Summary, get rid of the data that was saved for printing the previous time.

The following tables are used:
info_share, provider_info_share, placement_end

```
--*************************************************************************************
   PROCEDURE clean_up_action_summary
      (i_county_no       IN  info_share.county_no%TYPE,
      i_placement_id      IN  placement_end.placement_id%TYPE,
      i_log_id           IN  info_share.log_id%TYPE);

--*************************************************************************************************
```

■ ADD OR UPDATE WHO INFORMATION WAS SHARED WITH

This procedure saves some of the data for the Action Summary form. This is the information shared with new caseworker, new care giver, mother, father, the date it was shared, and how it was shared (by telephone, letter, or face to face).

The following tables are used:
info_share

```
--*************************************************************************************************
   PROCEDURE maintain_shared_info
      (i_county_no       IN  info_share.county_no%TYPE,
      i_log_id           IN  info_share.log_id%TYPE,
      i_info_share_no     IN  info_share.info_share_no%TYPE,
      i_shared_with_type  IN  info_share.shared_with_type%TYPE,
      i_date_shared       IN  VARCHAR2,
      i_how_shared        IN  info_share.how_shared%TYPE);

--*************************************************************************************
--     ADD OR UPDATE TYPE OF INFORMATION SHARED WITH NEW CAREGIVER
```

This procedure saves some of the data for the Action Summary form. This is the type of information shared with the new caregiver.

The following tables are used:
provider_info_share

```
--*************************************************************************************
   PROCEDURE maintain_prov_info_share
      (i_county_no       IN  provider_info_share.county_no%TYPE,
      i_placement_id      IN  provider_info_share.placement_id%TYPE,
      i_info_type_code    IN  VARCHAR2);

--*************************************************************************************
--     ADD PLACEMENT END REASONS
```

This procedure saves some of the data for the Action Summary form.

The following tables are used:
Activity, placement_end

```
--*************************************************************************************
```

```
   PROCEDURE placement_end_reasons
      (i_county_no       IN  placement_end.county_no%TYPE,
      i_placement_id     IN  placement_end.placement_id%TYPE,
      i_plcmnt_end_code  IN  VARCHAR2,
      i_other_reason     IN  placement_end.other_reason%TYPE,
      i_remarks          IN  placement_end.remarks%TYPE,
      i_rpt_type         IN  VARCHAR2,
      i_log_id           IN  activity.log_id%TYPE);

--*************************************************************************
```

■ update comment table

The following table is used:

■ comments

```
--*************************************************************************
   PROCEDURE update_action_comment
      (i_comment_id          IN  comments.comment_id%TYPE,
      i_county_no            IN  comments.county_no%TYPE,
      i_log_id               IN  comments.log_id%TYPE,
      i_comment_type         IN  comments.comment_type%TYPE,
      i_date_entered         IN  comments.date_entered%TYPE,
      i_text                 IN  comments.text%TYPE,
      i_entered_by_login_name IN  comments.entered_by_login_name%TYPE)
   IS
   BEGIN
      UPDATE comments
      SET text              =   i_text,
         date_entered       =   sysdate,
         entered_by_login_name = i_entered_by_login_name
      WHERE   comment_id    =   i_comment_id
      AND     county_no     =   i_county_no
      AND     log_id        =   i_log_id
      AND     comment_type  =   i_comment_type;

       EXCEPTION
          WHEN NO_DATA_FOUND THEN
            INSERT INTO comments
               (comment_id,
               county_no,
               log_id,
               comment_type,
               date_entered,
               text,
               entered_by_login_name)
            VALUES
               (i_comment_id,
               i_county_no,
               i_log_id,
               i_comment_type,
               sysdate,
               i_text,
               i_entered_by_login_name);
   END update_action_comment;

PACKAGE BODY      ACTION_SUMMARY_DATA
```

```
IS
--
-- Purpose: Retrieve action summary data for form/screen


--*******************************************************************************
--      RETRIEVE THE ITEMS FOR ACTION SUMMARY SCREEN
--*******************************************************************************
   PROCEDURE get_info_shared
      (i_county_no           IN  info_share.county_no%TYPE,
      i_log_id               IN  info_share.log_id%TYPE,
      GetInfoSharedCursor     IN OUT Info_Shared_Cursor)

   IS
   BEGIN
      OPEN GetInfoSharedCursor FOR
      SELECT  info_share_no,
            shared_with_type,
            TO_CHAR(date_shared,'MM/DD/YYYY') date_shared,
            how_shared
      FROM    info_share
      WHERE   county_no   = i_county_no
      AND     log_id      = i_log_id;

   END get_info_shared;

--*******************************************************************************
--  Retrieve placement_end data; action summary data for replacement or termination
--*******************************************************************************
   PROCEDURE get_plcmnt_end
      (i_county_no           IN  placement_end.county_no%TYPE,
      i_placement_id          IN  placement_end.placement_id%TYPE,
      PlcmntEndCursor         IN OUT Plcmnt_End_Cursor)

   IS
   BEGIN
      OPEN PlcmntEndCursor FOR
      SELECT plcmnt_end_code,
            remarks,
            other_reason
      FROM    placement_end
      WHERE   county_no       = i_county_no
      AND     placement_id    = i_placement_id
      ORDER BY plcmnt_end_code;

      /*EXCEPTION
         WHEN NO_DATA_FOUND THEN
            SELECT plcmnt_end_code,
                  remarks,
                  other_reason
            FROM    placement_end
            WHERE   county_no       = i_county_no
            AND     placement_id    = i_placement_id
            AND end_date = (SELECT MAX(end_date)
                        FROM placement_funding_source
```

```
                        WHERE county_no = i_county_no
                        AND placement_id = v_placement_id);
      END;

*/    END get_plcmnt_end;


--*************************************************************************
-- Retrieve action summary comments
--*************************************************************************
   PROCEDURE get_action_summary_comments
      (i_county_no          IN  placement_comment.county_no%TYPE,
      i_placement_id        IN  placement_comment.placement_id%TYPE,
      ActSumCommentsCursor IN OUT Act_Sum_Comments_Cursor)
   IS
   BEGIN
      OPEN ActSumCommentsCursor FOR
      SELECT  placement_comment.comment_id,
            comment_type
      FROM    placement_comment,comments
      WHERE   placement_comment.county_no     = i_county_no
      AND     placement_id                    = i_placement_id
      AND     comments.comment_id             = placement_comment.comment_id
      AND     comment_type IN ('P1','P2','P3','P4','P5','P6','P7')
      ORDER BY comment_type ASC;

   END get_action_summary_comments;


--*************************************************************************
-- Retrieve type of information shared with the new provider
--*************************************************************************
   PROCEDURE get_provider_info_share
      (i_county_no          IN  provider_info_share.county_no%TYPE,
      i_placement_id        IN  provider_info_share.placement_id%TYPE,
      ProvInfoShareCursor     IN OUT Prov_Info_Share_Cursor)

   IS
   BEGIN
      OPEN ProvInfoShareCursor FOR
      SELECT  info_type_code
      FROM    provider_infO_share
      WHERE   county_no       = i_county_no
      AND     placement_id    = i_placement_id
      ORDER BY info_type_code ASC;
   END get_provider_info_share;

--*************************************************************************
-- After printing form, clear activity table
--*************************************************************************
   PROCEDURE delete_activity
      (i_county_no          IN  activity.county_no%TYPE,
      i_log_id              IN  activity.log_id%TYPE)
   IS
   BEGIN
```

```
      DELETE FROM activity
      WHERE   county_no       =   i_county_no
      AND     log_id          =   i_log_id
      AND     type_code       IN  ('CR','FM','MM','CC','TP','CLOSE');
   END delete_activity;

--*****************************************************************************
--  Get current primary worker and the last primary worker
--*****************************************************************************
   PROCEDURE get_worker_data
      (i_county_no            IN  load_case.county_no%TYPE,
      i_log_id                IN  load_case.log_id%TYPE,
      current_wkr_name        OUT VARCHAR2,
      current_wkr_ph          OUT VARCHAR2,
      previous_wkr_name       OUT VARCHAR2,
      previous_wkr_ph         OUT VARCHAR2,
      current_load_no         OUT load_case.load_no%TYPE,
      previous_load_no        OUT load_case.load_no%TYPE)
   IS
      v_area_code     phone.area_code%TYPE;
      v_phone_no      phone.phone_no%TYPE;
      v_worker_id     worker.worker_id%type;
begin
   BEGIN
   select  primary_worker_id
   into v_worker_id
   from swss_case
   WHERE   county_no     = i_county_no
   AND     log_id        = i_log_id;

   select worker_name
   into  current_wkr_name
   from worker
   where worker_id= v_worker_id;

   select load_no
   into  current_load_no
   from load_case
   WHERE   load_case.county_no     = i_county_no
   AND     load_case.log_id        = i_log_id
   and     responsibility = 'P'
   and     end_date is null;

   --get current primary worker
   begin
      SELECT  area_code,
              phone_no

      INTO v_area_code,
         v_phone_no
      FROM    phone
      WHERE   phone_id =  (select phone_id
                           from worker_phone
                           where worker_id = v_worker_id
                           AND     phone_type_code = 'P1');
```

```
      IF NOT v_phone_no IS NULL then
         current_wkr_ph := '(' || v_area_code || ') ' ||
                           SUBSTR(v_phone_no,1,3) || '-' ||
                           SUBSTR(v_phone_no,4,4);
      END IF;
exception
when others then
null;
end;
/*       --get previous primary worker
     begin
      SELECT worker_name,
            area_code,
            phone_no,
            load_case.load_no
      INTO previous_wkr_name,
         v_area_code,
         v_phone_no,
         previous_load_no
      FROM load_case,load,worker,worker_phone,phone
      WHERE   load_case.county_no     = i_county_no
      AND     load_case.log_id        = i_log_id
      AND     responsibility          = 'P'
      AND     end_date                = (SELECT MAX(end_date)
                                         FROM load_case
                                         WHERE county_no = i_county_no
                                         AND     log_id  = i_log_id
                                         AND     responsibility = 'P'
                                         AND     NOT end_date IS NULL)
      AND     load.load_no            = load_case.load_no
      AND     worker.worker_id        = load.worker_no
      AND     worker_phone.worker_id(+) = worker.worker_id
      AND     worker_phone.phone_type_code(+) = 'P1'
      AND     phone.phone_id(+)       = worker_phone.phone_id;*/

      exception
      when no_data_found then
      null;
      when others then
      null;
      end;


   END get_worker_data;

--*************************************************************************************
--  GET Current provider id and previous provider id
--*************************************************************************************
   PROCEDURE find_provider_data
      (i_county_no           IN provider_case_person.county_no%TYPE,
      i_log_id               IN  provider_case_person.log_id%TYPE,
      i_person_id            IN  provider_case_person.person_id%TYPE,
      current_placement_id    OUT provider_case_person.placement_id%TYPE,
      current_provider_id     OUT provider_case_person.provider_id%TYPE,
```

```
    current_prov_type       OUT provider_case_person.provider_type%TYPE,
    previous_placement_id   OUT provider_case_person.placement_id%TYPE,
    previous_provider_id    OUT provider_case_person.provider_id%TYPE,
    previous_prov_type      OUT provider_case_person.provider_type%TYPE,
    current_funding_source  OUT placement_funding_source.funding_source_code%TYPE,
    previous_funding_source OUT placement_funding_source.funding_source_code%TYPE)
  IS
    v_placement_id provider_case_person.placement_id%TYPE;
  BEGIN
    BEGIN
    SELECT  placement_id,
            provider_id,
            provider_type
    INTO    v_placement_id,
            current_provider_id,
            current_prov_type
    FROM    provider_case_person
    WHERE   county_no       = i_county_no
    AND     log_id          = i_log_id
    AND     person_id       = i_person_id
    AND     prov_end_date IS NULL;

    EXCEPTION
      WHEN NO_DATA_FOUND THEN
      BEGIN
      SELECT  placement_id,
              placing_agency_id,
              placing_agency_type
      INTO    current_placement_id,
              current_provider_id,
              current_prov_type
      FROM    provider_case_person
      WHERE   county_no       = i_county_no
      AND     log_id          = i_log_id
      AND     person_id       = i_person_id
      AND     prov_end_date IS NULL;

      EXCEPTION
        WHEN NO_DATA_FOUND THEN
        current_placement_id := NULL;
        v_placement_id := NULL;
      END;
    END;

    current_placement_id := v_placement_id;


    --find previous provider
    BEGIN
    SELECT  placement_id,
            provider_id,
            provider_type
    INTO    v_placement_id,
            previous_provider_id,
            previous_prov_type
```

```
      FROM    provider_case_person
      WHERE   county_no       = i_county_no
      AND     log_id          = i_log_id
      AND     person_id       = i_person_id
      AND     prov_end_date   = (SELECT MAX(prov_end_date)
                        FROM provider_case_person
                        WHERE county_no     = i_county_no
                        AND     log_id      = i_log_id
                        AND     person_id   = i_person_id
                        AND     NOT prov_end_date IS NULL);

      EXCEPTION
      WHEN NO_DATA_FOUND THEN
         BEGIN
            SELECT  placement_id,
                    placing_agency_id,
                    placing_agency_type
            INTO    previous_placement_id,
                    previous_provider_id,
                    previous_prov_type
            FROM    provider_case_person
            WHERE   county_no       = i_county_no
            AND     log_id          = i_log_id
            AND     person_id       = i_person_id
            AND     prov_end_date   = (SELECT MAX(prov_end_date)
                              FROM provider_case_person
                              WHERE county_no     = i_county_no
                              AND     log_id      = i_log_id
                              AND     person_id   = i_person_id
                              AND     NOT prov_end_date IS NULL);
               EXCEPTION
            WHEN NO_DATA_FOUND THEN
               v_placement_id := NULL;
               previous_placement_id := NULL;
         END;
      previous_placement_id := v_placement_id;
      END;
--Find funding source code for the placement
      BEGIN
         IF v_placement_id IS NULL THEN
            previous_funding_source := NULL;
         ELSE
            SELECT funding_source_code
            INTO    previous_funding_source
            FROM    placement_funding_source
            WHERE   county_no = i_county_no
            AND     placement_id = v_placement_id
            AND     end_date IS NULL;
         END IF;
      EXCEPTION
         WHEN NO_DATA_FOUND THEN
         BEGIN
            SELECT funding_source_code
            INTO previous_funding_source
            FROM placement_funding_source
```

```
            WHERE county_no = i_county_no
            AND placement_id = v_placement_id
            AND end_date = (SELECT MAX(end_date)
                        FROM placement_funding_source
                        WHERE county_no = i_county_no
                        AND placement_id = v_placement_id
                        AND NOT end_date IS NULL);
         EXCEPTION
            WHEN NO_DATA_FOUND THEN
               previous_funding_source := NULL;
         END;
      END;

   END find_provider_data;

--*************************************************************************************
-- GET latest docket number
--*************************************************************************************
   PROCEDURE find_docket_number
      (i_county_no          IN  petition_case_person.county_no%TYPE,
      i_log_id              IN  petition_case_person.log_id%TYPE,
      i_person_id           IN  petition_case_person.person_id%TYPE,
      o_docket_no           OUT court_petition.docket_no%TYPE)
   IS
   BEGIN
   begin
      SELECT  DISTINCT(docket_no)
      INTO    o_docket_no
      FROM    petition_case_person, petition, court_petition
      WHERE   petition_case_person.county_no      = i_county_no
      AND     petition_case_person.log_id         = i_log_id
      AND     petition_case_person.person_id      = i_person_id
      AND     petition.petition_id                = petition_case_person.petition_id
      AND     petition_date                       = (SELECT MAX(petition_date)
                                                  FROM petition,petition_case_person
                                                  WHERE petition_case_person.county_no =
                                                        i_county_no
                                                  AND   log_id = i_log_id
                                                  AND   person_id = i_person_id
                                                  AND   petition.petition_id =
                                                        petition_case_person.petition_id)
      AND     court_petition.petition_id          = petition.petition_id;
         EXCEPTION
         WHEN NO_DATA_FOUND THEN
            o_docket_no:=NULL;
           end;
   END find_docket_number;

--*************************************************************************************
-- GET which action summary types need printing
--*************************************************************************************
   PROCEDURE get_rept_types
      (i_county_no          IN  activity.county_no%TYPE,
      i_log_id              IN  activity.log_id%TYPE,
      GetReptTypesCursor    IN OUT Get_Rept_Types_Cursor)
```

```
   IS
   BEGIN
      OPEN GetReptTypesCursor FOR
      SELECT type_code
      FROM    activity
      WHERE   county_no       = i_county_no
      AND     log_id          = i_log_id
      AND     type_code IN ('FM','MM','CC','CR','TP','CLOSE');
   END get_rept_types;


/*-------------------------------------------------------------------------------------------------------
PROCEDURE get_current_placement
                  (i_person_id                IN      provider_case_person.person_id%TYPE,
                  i_log_id                   IN        provider_case_person.log_id%TYPE,
                  i_county_no                 IN      provider_case_person.county_no%TYPE,
      o_placement_id              OUT    provider_case_person.placement_id%TYPE)
IS

BEGIN
 begin
         SELECT  placement_id
   into    o_placement_id
   FROM    provider_case_person
         WHERE          county_no = i_county_no
           AND log_id    = i_log_id
           AND person_id = i_person_id
           AND prov_end_date IS NULL;

   exception
   when no_data_found then

begin
   SELECT  placement_id
   into    o_placement_id
   FROM    provider_case_person
         WHERE          county_no = i_county_no
           AND log_id    = i_log_id
           AND person_id = i_person_id
     AND  prov_end_date = (SELECT MAX(prov_end_date)
                           FROM provider_case_person
                           WHERE county_no = i_county_no
                           AND log_id = i_log_id
                           and person_id = i_person_id);
                     EXCEPTION
              WHEN NO_DATA_FOUND THEN
                 o_placement_id:= NULL;
      end;
   end;
end get_current_placement;

PROCEDURE get_child_case_info
         (i_county                 IN swss_case.county_no%TYPE,
         i_log_id                  IN swss_case.log_id%TYPE,
```

```
        o_child_dob             OUT VARCHAR2,
        o_child_sex             OUT group_person.sex%TYPE,
        o_child_place_date      OUT VARCHAR2,
        o_child_recip_id        OUT group_person.recipient_id%TYPE,
        o_child_race_code       OUT race.race_code%TYPE)
IS
        v_last_name    VARCHAR2(19);
        v_first_name   VARCHAR2(12);
   v_child_id      swss_case.child_id%TYPE;
   v_sib_grp_id    swss_case.sib_grp_id%TYPE;
BEGIN
 BEGIN
--  GET CHILDS INFORMATION
        SELECT     sex,
                   to_char(date_of_birth, 'YYYYMMDD'),
           recipient_id,
           swss_case.sib_grp_id,
           child_id
        INTO       o_child_sex,
                   o_child_dob,
                   o_child_recip_id,
                   v_sib_grp_id,
           v_child_id
        FROM  swss_case, group_person
        WHERE          swss_case.county_no = i_county
                and swss_case.log_id = i_log_id
                and group_person.county_no = i_county
                and group_person.person_id = swss_case.child_id
                and group_person.sib_grp_id = swss_case.sib_grp_id;
 EXCEPTION
        WHEN NO_DATA_FOUND
                THEN RAISE_APPLICATION_ERROR
                        (-20700, ' Case/Child not found-(get_child_case_info)');
 END;
 BEGIN
--  GET CHILDS PRIMARY RACE
        SELECT     race_code

        INTO       o_child_race_code

        FROM  group_person_race
        WHERE          group_person_race.county_no = i_county
                and group_person_race.sib_grp_id = v_sib_grp_id
                and group_person_race.person_id = v_child_id
      and group_person_race.race_order_no = 1;
 EXCEPTION
        WHEN NO_DATA_FOUND
                THEN null;
 END;


 BEGIN
        -- Get current adoption placement date
        SELECT         TO_CHAR(prov_begin_date,'MM/DD/YYYY')
        INTO    o_child_place_date
```

```
        FROM  provider_case_person
        WHERE         county_no = i_county
      and living_arrangement_code = '04'
                and prov_begin_date =         (SELECT max(prov_begin_date)
                                      FROM  provider_case_person
                                      WHERE         county_no = i_county
                                              and log_id = i_log_id
                                              and person_id = v_child_id
                 and logical_delete is null)
                and log_id = i_log_id
                and person_id = v_child_id
      and logical_delete is null;

  EXCEPTION
                WHEN NO_DATA_FOUND THEN
      o_child_place_date := null;
  END;

END get_child_case_info;

PROCEDURE get_mother_addr_hist
        (i_county                 IN swss_case.county_no%TYPE,
   i_log_id       IN swss_case.log_id%TYPE,
   GetAddrCursor   IN OUT Get_Addr_Cursor)

IS
   v_child_id      swss_case.child_id%TYPE;
        v_sib_grp_id    group_person.sib_grp_id%TYPE;
        v_parents_id   group_person.person_id%TYPE;
   v_area_code     phone.area_code%TYPE;
   v_phone_no      phone.phone_no%TYPE;
   v_foreign_phone VARCHAR(30);
   v_symbol1       VARCHAR(1);
   v_symbol2       VARCHAR(2);
   v_symbol3       VARCHAR(1);
   v_zip           VARCHAR(10);
   v_name          VARCHAR(90);
   v_unknown_address   VARCHAR(1);
   v_char_left     VARCHAR(60);
   v_address_line VARCHAR(60);
   v_province          VARCHAR(10);
   v_country           VARCHAR(10);

BEGIN
-- Get the child's id and sib group id
   SELECT  child_id, sib_grp_id
   INTO    v_child_id, v_sib_grp_id
   FROM    swss_case
   WHERE   county_no = i_county
        and log_id = i_log_id;

-- Get the bio mom data
BEGIN
   SELECT person1_id
   INTO    v_parents_id
```

```
   FROM    relation,group_person
   WHERE   relation.county_no         = i_county
   AND     relation.log_id            = i_log_id
   AND     relation.person2_id        = v_child_id
   AND     relation.relation_type     = 'BP'
   AND     group_person.county_no     = relation.county_no
   AND     group_person.sib_grp_id    = v_sib_grp_id
   AND     group_person.person_id     = relation.person1_id
   AND     group_person.sex           = 'F';
 EXCEPTION
         WHEN NO_DATA_FOUND
                THEN null;
END;

--  Get all of the addresses for this person in the case
--  Most current address first (ORDER BY desc)
   OPEN GetAddrCursor FOR
      SELECT  SUBSTR(last_name,1,25)||', '||SUBSTR(first_name,1,20) name,
--              unknown_address,
--              address.country,
--              address.province,
--              address.address_id,
--The DECODE checks if unknown address, send back UNKNOWN for the street address
--If the country is USA, send back line1 of the address
--If the country is not USA, string together line1, province and country
             DECODE(group_person.unknown_address,'Y','Unknown',
                (DECODE(address.country,'USA',address.line1,
                SUBSTR(address.line1,1,(60 - LENGTH(province) - LENGTH(country) - 6))||
                '/'||SUBSTR(province,1,10)||
                '/'||SUBSTR(country,1,10))))
                address_line_1,
             address.city,
                  address.state_code state,
             address.zip || address.zip_plus zip,
             TO_CHAR(group_person_address.addr_date, 'mm/dd/yyyy') change_date,
             GetPhone(i_county,v_sib_grp_id,person1_id) phone_no
                  FROM  relation,group_person,address, group_person_address
      WHERE   relation.county_no = i_county
             and relation.log_id = i_log_id
             and relation.person2_id = v_child_id
             and relation_type = 'BP'
             and group_person.county_no = relation.county_no
             and group_person.sib_grp_id = v_sib_grp_id
             and group_person.person_id = relation.person1_id
             and group_person.sex = 'F'
             and address.address_id IN (SELECT address_id
                                                FROM   group_person_address
                                                WHERE  county_no = i_county
                        and sib_grp_id = v_sib_grp_id
                        and person_id = v_parents_id
                        and addr_type_code = 'P')
             and group_person_address.address_id = address.address_id
             ORDER BY addr_date desc;
 EXCEPTION
         WHEN NO_DATA_FOUND
```

```
                THEN null;

END get_mother_addr_hist;

--*************************************************************************
-- 10/28/1999 BJC changed for Merant driver
--*************************************************************************
PROCEDURE get_father_addr_hist
        (i_county                IN swss_case.county_no%TYPE,
   i_log_id       IN swss_case.log_id%TYPE,
   GetAddrCursor   IN OUT Get_Addr_Cursor)
IS
   v_child_id      swss_case.child_id%TYPE;
        v_sib_grp_id    group_person.sib_grp_id%TYPE;
        v_parents_id    group_person.person_id%TYPE;
   v_area_code     phone.area_code%TYPE;
   v_phone_no      phone.phone_no%TYPE;
   v_foreign_phone VARCHAR(30);
   v_symbol1       VARCHAR(1);
   v_symbol2       VARCHAR(2);
   v_symbol3       VARCHAR(1);
   v_zip           VARCHAR(10);
   v_name          VARCHAR(90);
   v_unknown_address   VARCHAR(1);
   v_char_left     VARCHAR(60);
   v_address_line VARCHAR(60);
   v_province          VARCHAR(10);
   v_country           VARCHAR(10);
BEGIN
-- Get the child's id and sib group id
   SELECT  child_id, sib_grp_id
   INTO    v_child_id, v_sib_grp_id
   FROM    swss_case
   WHERE   county_no = i_county
        and log_id = i_log_id;
-- Get the bio dad data
BEGIN
   SELECT person1_id
   INTO    v_parents_id
   FROM    relation,group_person
   WHERE   relation.county_no          = i_county
   AND     relation.log_id             = i_log_id
   AND     relation.person2_id         = v_child_id
   AND     relation.relation_type      = 'BP'
   AND     group_person.county_no      = relation.county_no
   AND     group_person.sib_grp_id     = v_sib_grp_id
   AND     group_person.person_id      = relation.person1_id
   AND     group_person.sex            = 'M';
 EXCEPTION
        WHEN NO_DATA_FOUND
                THEN null;
END;

--  Get all of the addresses for this person in the case
--  Most current address first (ORDER BY desc)
```

```
   OPEN GetAddrCursor FOR
      SELECT  SUBSTR(last_name,1,25)||', '||SUBSTR(first_name,1,20) name,
--              unknown_address,
--              address.country,
--              address.province,
--              address.address_id,
--The DECODE checks if unknown address, send back UNKNOWN for the street address
--If the country is USA, send back line1 of the address
--If the country is not USA, string together line1, province and country
             DECODE(group_person.unknown_address,'Y','Unknown',
                (DECODE(address.country,'USA',address.line1,
                SUBSTR(address.line1,1,(60 - LENGTH(province) - LENGTH(country) - 6))||
                '/'||SUBSTR(province,1,10)||
                '/'||SUBSTR(country,1,10))))
                address_line_1,
             address.city,
                  address.state_code state,
             address.zip || address.zip_plus zip,
             TO_CHAR(group_person_address.addr_date, 'mm/dd/yyyy') change_date,
             GetPhone(i_county,v_sib_grp_id,person1_id) phone_no
                  FROM  relation,group_person,address, group_person_address
      WHERE   relation.county_no = i_county
             and relation.log_id = i_log_id
             and relation.person2_id = v_child_id
             and relation_type = 'BP'
             and group_person.county_no = relation.county_no
             and group_person.sib_grp_id = v_sib_grp_id
             and group_person.person_id = relation.person1_id
             and group_person.sex = 'M'
             and address.address_id IN (SELECT address_id
                                                FROM   group_person_address
                                                WHERE  county_no = i_county
                        and sib_grp_id = v_sib_grp_id
                        and person_id = v_parents_id
                        and addr_type_code = 'P')
             and group_person_address.address_id = address.address_id
             ORDER BY addr_date desc;
 EXCEPTION
         WHEN NO_DATA_FOUND
                  THEN null;
END get_father_addr_hist;


-- ****************************************************************************
-- **  PROCEDURE: INSERT_REC
-- **
-- **  PURPOSE:  TO INSERT A RECORD IN THE COMMENTS TABLE.
-- **
-- ****************************************************************************
PROCEDURE INSERT_REC(I_COMMENT_TYPE IN  COMMENTS.COMMENT_TYPE%TYPE,
I_TEXT        IN  COMMENTS.TEXT%TYPE,
I_COUNTY_NO    IN  COMMENTS.COUNTY_NO%TYPE,
I_LOG_ID      IN  COMMENTS.LOG_ID%TYPE,
O_COMMENT_ID   OUT COMMENTS.COMMENT_ID%TYPE)
IS
```

```
V_COMMENT_ID  COMMENTS.COMMENT_ID%TYPE;
V_USERID     COMMENTS.ENTERED_BY_LOGIN_NAME%TYPE;
BEGIN
IF LENGTH(I_TEXT) > FIND_MAX_TYPE_LENGTH(I_COMMENT_TYPE) THEN
RAISE TEXT_TOO_LONG;
ELSE
SELECT COMMENT_ID.NEXTVAL INTO V_COMMENT_ID FROM DUAL;
o_comment_id := v_comment_id;
V_USERID := FIND_DB_USER;
DBMS_OUTPUT.PUT_LINE(TO_CHAR(V_COMMENT_ID));
DBMS_OUTPUT.PUT_LINE(V_USERID);
INSERT INTO COMMENTS(COMMENT_ID, COMMENT_TYPE, DATE_ENTERED
   ,TEXT, ENTERED_BY_LOGIN_NAME, COUNTY_NO, LOG_ID
   ,MODIFIED_DATE, MODIFIED_BY_LOGIN_NAME)
VALUES  (V_COMMENT_ID, I_COMMENT_TYPE, SYSDATE
   ,I_TEXT, V_USERID, I_COUNTY_NO,I_LOG_ID
   ,SYSDATE, V_USERID);
--COMMIT;
END IF;
EXCEPTION
WHEN TEXT_TOO_LONG THEN
RAISE_APPLICATION_ERROR(-20800, 'The text value exceeds the maximum length.');
END INSERT_REC;

**************************************************************************
-GET Social Work Contacts
**************************************************************************

PROCEDURE GET_CONTACTS(
                LOGID           IN SWSS.CONTACT.LOG_ID%TYPE,
                COUNTY_CODE     IN SWSS.CONTACT.COUNTY_NO%TYPE,
                CONTACTS_CUR    IN OUT contacts.get_contacts_cur
                )

 IS

 BEGIN

 OPEN CONTACTS_CUR FOR

 SELECT

    C.COUNTY_NO
   ,C.CONTACT_SEQ_NO
   ,C.LOG_ID
   ,C.CONTACT_METHOD
   ,C.CONTACT_DATE
   ,C.WORKER_ID
   ,C.SCHEDULED_IND
   ,C.KEPT_IND
   ,C.CONTACT_NOTES_ID
   ,C.SUMMARY_COMMENT_ID
   ,C.CONTACT_LOCATION
   ,C.CONTACT_METHOD_CD
   ,contacts.GET_METHOD(c.contact_method_cd) METHOD
```

```
   ,contacts.Get_NAME(c.contact_seq_no)NAME
   ,S.COMMENT_ID
   ,GET_NOTE(C.CONTACT_NOTES_ID) NOTE
   ,GET_SUMMARY(C.SUMMARY_COMMENT_ID) SUMMARY
   ,S.DATE_ENTERED

FROM

     SWSS.CONTACT C
     /*,SWSS.CONTACT_NAME N*/
     ,SWSS.COMMENTS S
     /*,SWSS.COMMENTS S1*/

WHERE

     C.COUNTY_NO =  COUNTY_CODE
         AND
      C.LOG_ID = LOGID
      /*  AND
      C.CONTACT_SEQ_NO = N.CONTACT_SEQ_NO*/
         AND
      C.SUMMARY_COMMENT_ID = S.COMMENT_ID
      ORDER BY C.CONTACT_DATE DESC;

/*exception
   when others then
      --null;  --
      dbms_output.put_line('SQL Code='||TO_CHAR(SQLCODE)); -- Typical usage
      dbms_output.put_line(sqlerrm);*/
END GET_CONTACTS;


--*********************************************************************
function GetAcceptDate
--
--Given a case county and the log id, return the Open date, which may be null
--
(i_CaseCounty in swss_case.county_no%type,
 i_LogID in swss_case.log_id%type)
 return date
is
   v_AcceptDate date;
begin
   select  ACCEPT_DATE
   into    v_AcceptDate
   from    swss_case
   where   county_no = i_CaseCounty
   and     log_id = i_LogID;
   return v_AcceptDate;

end GetAcceptDate;
```

1.3.11 Data Warehouse

If known, explain which items are added to the data warehouse and under what conditions they are written. Hopefully we can reference a document or set of documents supported by the data warehouse.

Placement data is written to the warehouse. Database is taking care of that part of the SWSS application – let them fill this out.

No fields in Action Summary Module are sent to the warehouse.

1.3.12 Technical Issues

Discuss any tricky things that happen in the module that someone who maintains the application may not recognize at first glance. Sibling group sharing, legal status switches, or reusing person IDs.

None for Action Summary Module.

1.3.13 Test Plans

Include the test plan developed for this module, and references to any scenarios that apply to it.

TEST PLAN

ACTION SUMMARY

General

1. One of five events must occur prior to entrance into Action Summary; 1) Case closing, 2) Termination from Foster Care Placement, 3) Child Replacement, 4) Caseworker change, and 5) Parent move.
2. If none of the above conditions are met, the user is notified via a message box then returned to the Report Generation menu.
3. After any one of the above conditions is met, the user is given a choice of entering the Action Summary module or returning to it later.

Specific Edits and Screen Displays

1. All dates entered must be valid.
2. If the reason for printing the Action Summary Report is Termination, the following is required; 1) reason for termination, 2) indication if medical information was given to the parents and if yes, a description of the medical information, 3) summary of services that were provided, 4) summary of services currently being provided and 5) summary of services and needs that still need to be met.
3. If the reason for printing the Action Summary Report is Closing, the following is required; 1) reason for termination from foster care, 2) information was shared with, 3) Social Work Contacts Report Period covered, 4) Reason for Closure, 5) Services that were provided, 6) Services currently provided, 7) Services and needs that still need to be met, 8) Was medical info given to parents, 9) Termination or closure explained to all parties, and 10) If termination unplanned, summarize reasons.
4. If the reason for completing is Case Closure, Social Work Contacts will print at the end of the Report (if there are any contacts to print after the time period specified).